Leith Price T1A3

Terminal Assessment Presentation

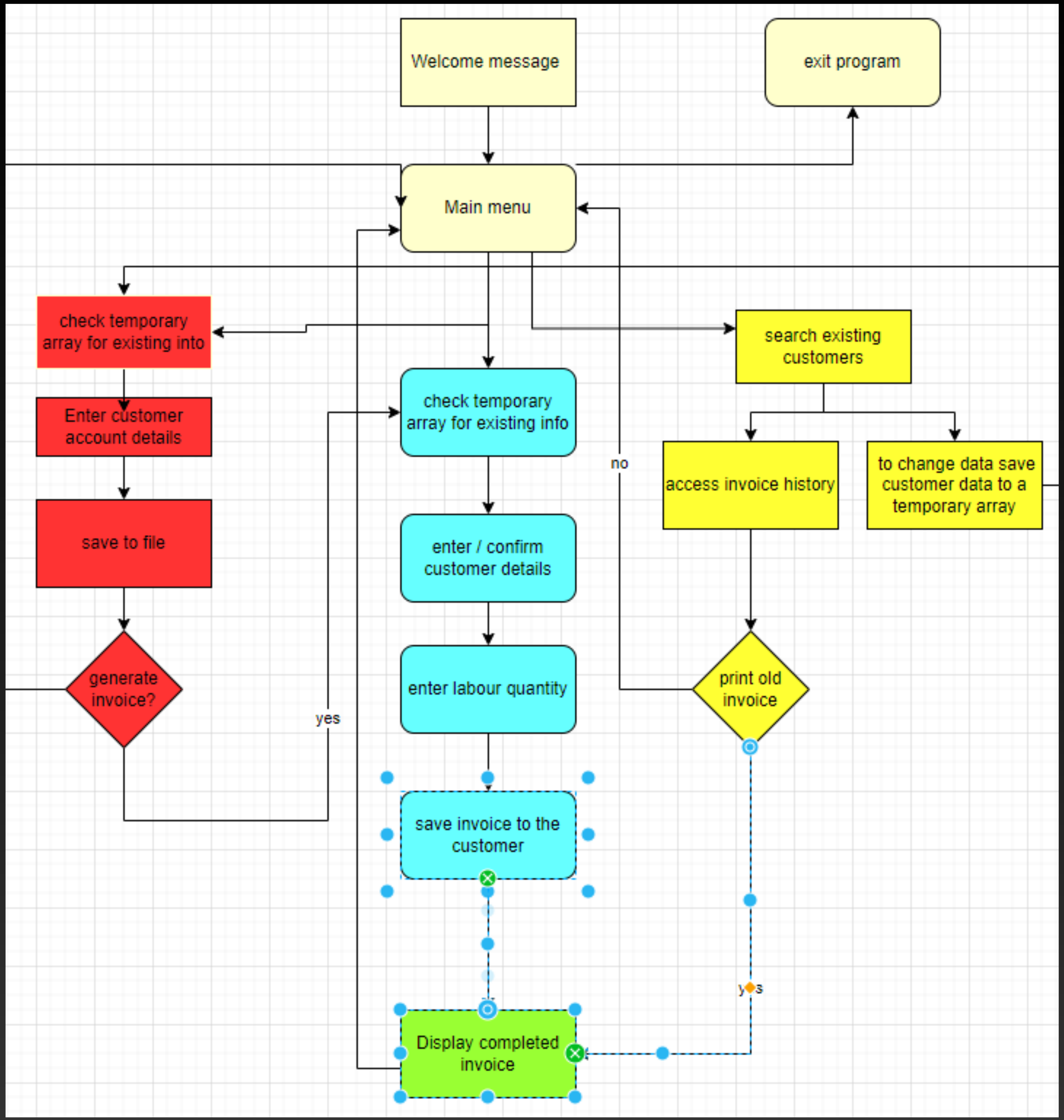
Project Plan

- Objective
- Flow chart
- Workflow planning
- Hard coding
- Features

Program Objective

The goal of the project was to create a Workshop program that would allow the users, workshop operators, to enter customer details and generate sales invoices.

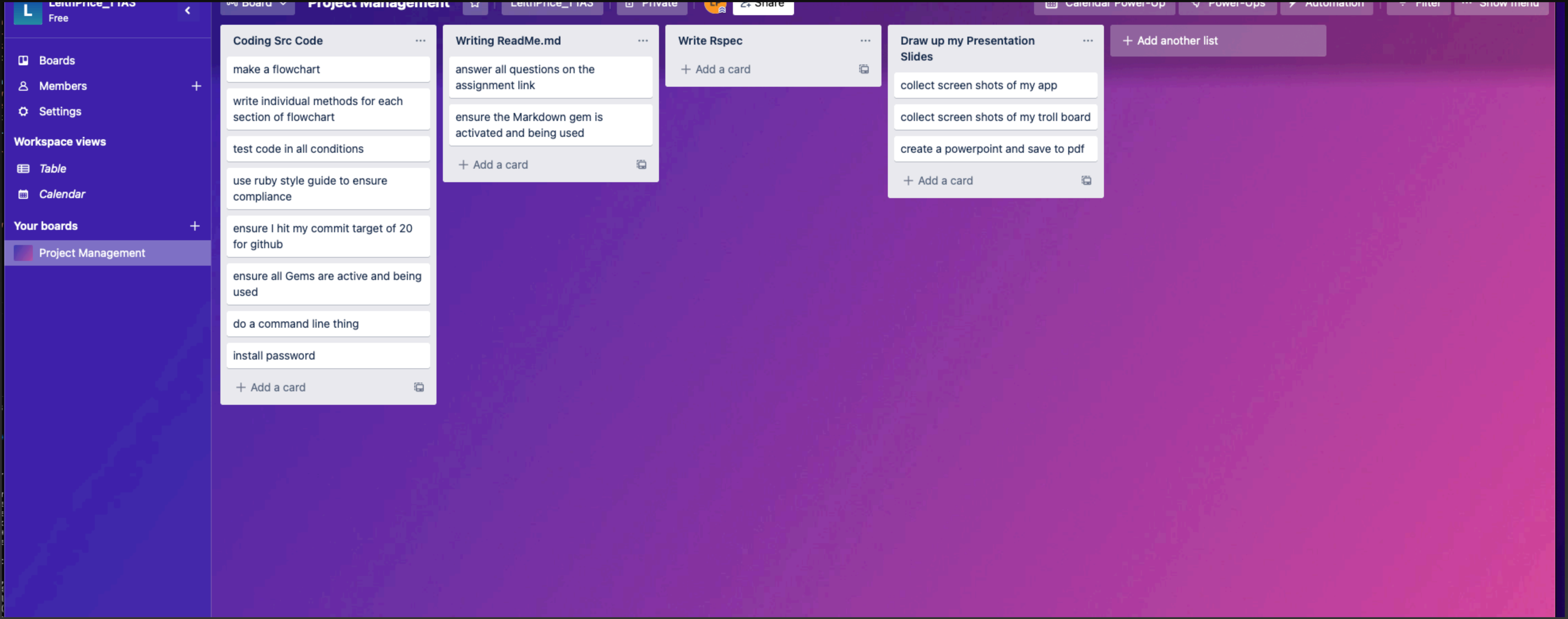
To reduce doubling up of user workload the App was designed to save user input into a .txt file so it could be accessed at later times when wanting to generate further invoices.


Welcome message
exit program
Main menu
check temporary array for existing into
Enter customer account details
save to file
generate invoice?
yes
check temporary array for existing info
enter / confirm customer details
enter labour quantity
save invoice to the customer
Display completed invoice
search existing customers
access invoice history
to change data save customer data to a temporary array
print old invoice
no
yes

Flowchart

Workflow planning

Trello Board



Coding Src Code

- make a flowchart
- write individual methods for each section of flowchart
- test code in all conditions
- use ruby style guide to ensure compliance
- ensure I hit my commit target of 20 for github
- ensure all Gems are active and being used
- do a command line thing
- install password

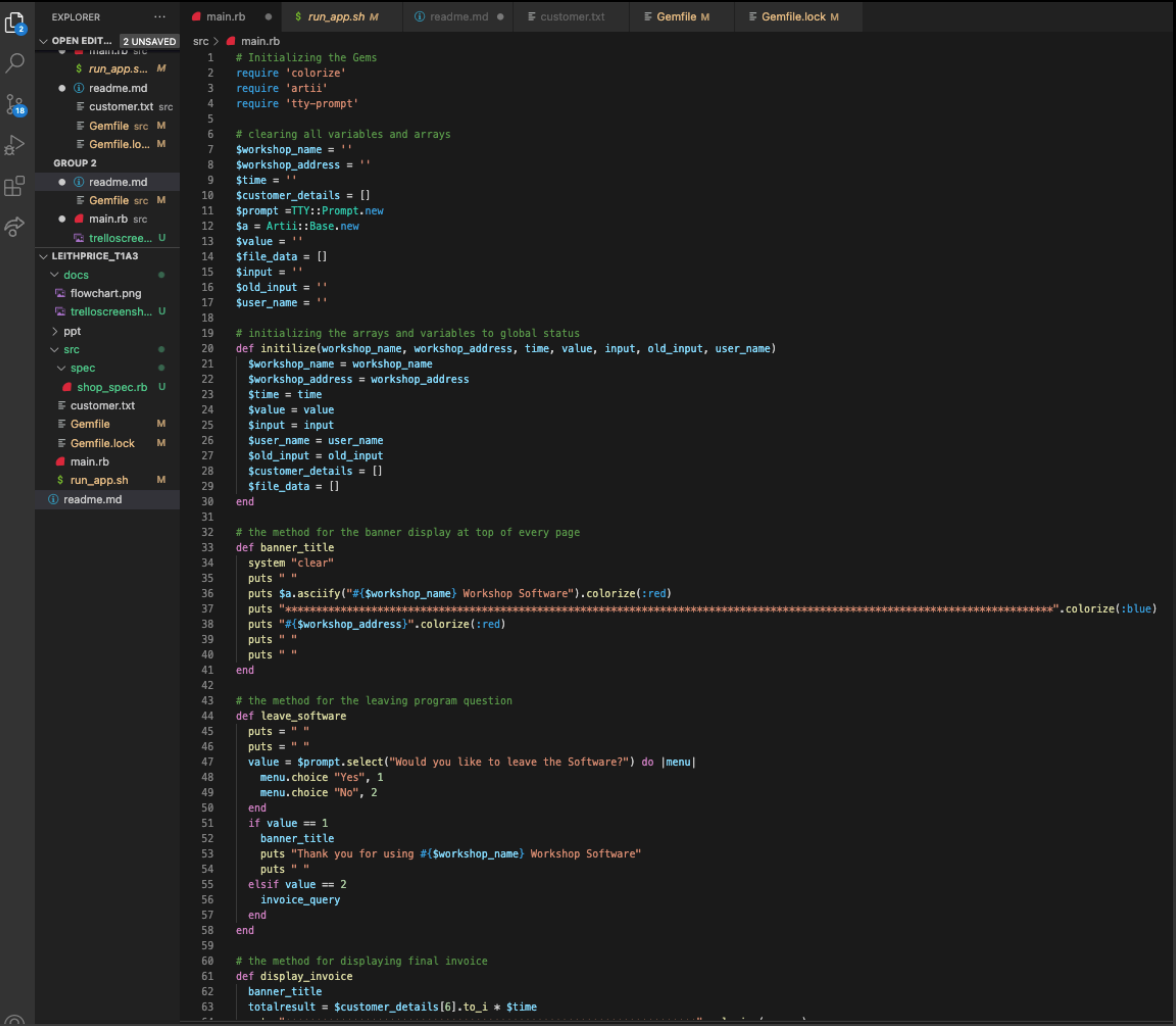
Writing ReadMe.md

- answer all questions on the assignment link
- ensure the Markdown gem is activated and being used

Write Rspec

Draw up my Presentation Slides

- collect screen shots of my app
- collect screen shots of my troll board
- create a powerpoint and save to pdf



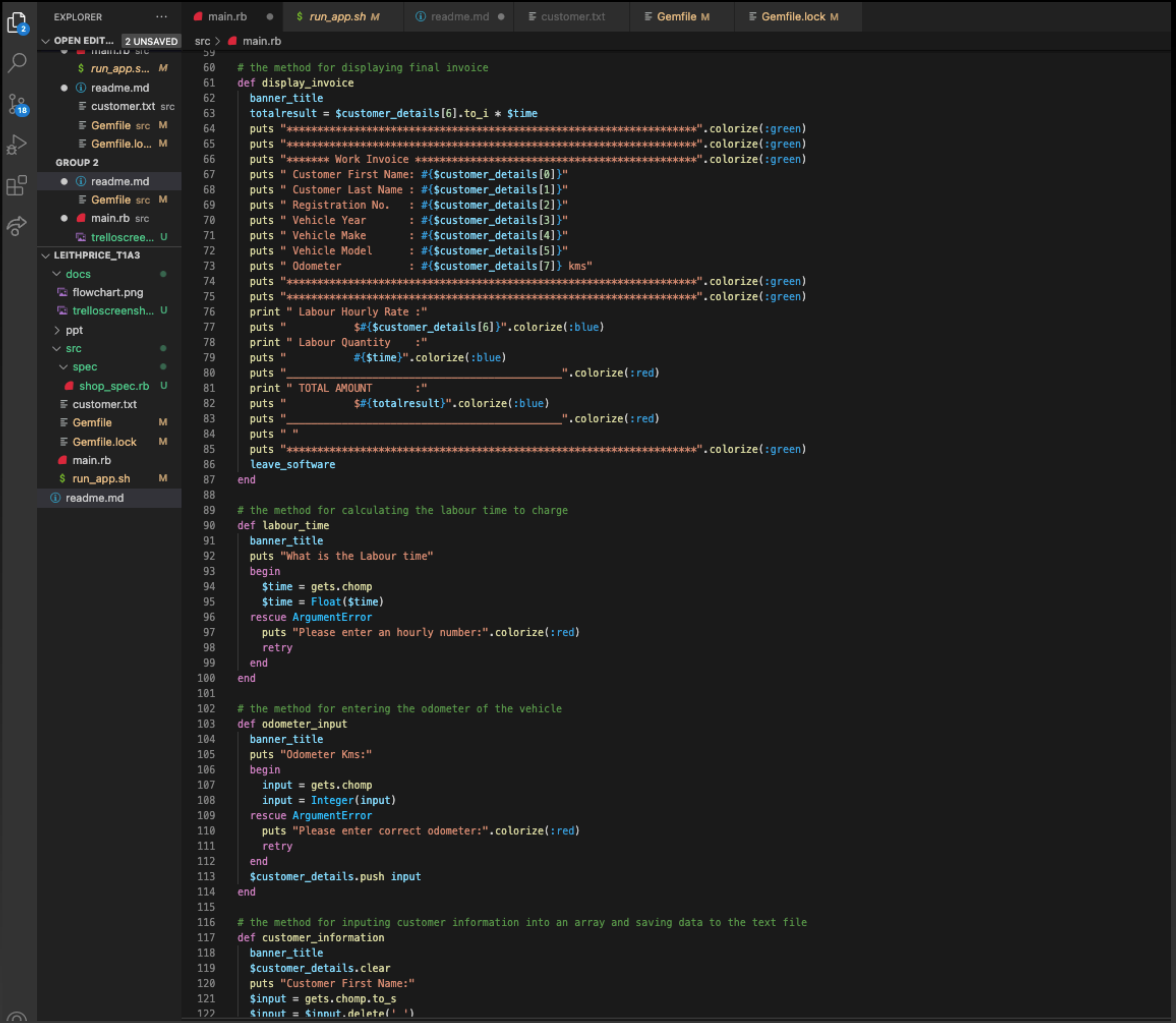
```ruby
# Initializing the Gems
require 'colorize'
require 'artii'
require 'tty-prompt'

# clearing all variables and arrays
$workshop_name = ''
$workshop_address = ''
$time = ''
$customer_details = []
$prompt =TTY::Prompt.new
$a = Artii::Base.new
$value = ''
$file_data = []
$input = ''
$old_input = ''
$user_name = ''

# initializing the arrays and variables to global status
def initilize(workshop_name, workshop_address, time, value, input, old_input, user_name)
  $workshop_name = workshop_name
  $workshop_address = workshop_address
  $time = time
  $value = value
  $input = input
  $user_name = user_name
  $old_input = old_input
  $customer_details = []
  $file_data = []
end

# the method for the banner display at top of every page
def banner_title
  system "clear"
  puts " "
  puts $a.asciify("#{$workshop_name} Workshop Software").colorize(:red)
  puts "*****************************************************************************************************************************".colorize(:blue)
  puts "#{$workshop_address}".colorize(:red)
  puts " "
  puts " "
end

# the method for the leaving program question
def leave_software
  puts = " "
  puts = " "
  value = $prompt.select("Would you like to leave the Software?") do |menu|
    menu.choice "Yes", 1
    menu.choice "No", 2
  end
  if value == 1
    banner_title
    puts "Thank you for using #{$workshop_name} Workshop Software"
    puts " "
  elsif value == 2
    invoice_query
  end
end

# the method for displaying final invoice
def display_invoice
  banner_title
  totalresult = $customer_details[6].to_i * $time
```

Program was made up of multiple methods.

Methods for

Banner Heading

Leaving the program



```ruby
# the method for displaying final invoice
def display_invoice
  banner_title
  totalresult = $customer_details[6].to_i * $time
  puts "*******************************************************************".colorize(:green)
  puts "*******************************************************************".colorize(:green)
  puts "******* Work Invoice **********************************************".colorize(:green)
  puts " Customer First Name: #{$customer_details[0]}"
  puts " Customer Last Name : #{$customer_details[1]}"
  puts " Registration No.   : #{$customer_details[2]}"
  puts " Vehicle Year       : #{$customer_details[3]}"
  puts " Vehicle Make       : #{$customer_details[4]}"
  puts " Vehicle Model      : #{$customer_details[5]}"
  puts " Odometer           : #{$customer_details[7]} kms"
  puts "*******************************************************************".colorize(:green)
  puts "*******************************************************************".colorize(:green)
  print " Labour Hourly Rate :"
  puts "           $#{$customer_details[6]}".colorize(:blue)
  print " Labour Quantity    :"
  puts "           #{$time}".colorize(:blue)
  puts "_____________________________________________".colorize(:red)
  print " TOTAL AMOUNT       :"
  puts "           $#{totalresult}".colorize(:blue)
  puts "_____________________________________________".colorize(:red)
  puts " "
  puts "*******************************************************************".colorize(:green)
  leave_software
end

# the method for calculating the labour time to charge
def labour_time
  banner_title
  puts "What is the Labour time"
  begin
    $time = gets.chomp
    $time = Float($time)
  rescue ArgumentError
    puts "Please enter an hourly number:".colorize(:red)
    retry
  end
end

# the method for entering the odometer of the vehicle
def odometer_input
  banner_title
  puts "Odometer Kms:"
  begin
    input = gets.chomp
    input = Integer(input)
  rescue ArgumentError
    puts "Please enter correct odometer:".colorize(:red)
    retry
  end
  $customer_details.push input
end

# the method for inputing customer information into an array and saving data to the text file
def customer_information
  banner_title
  $customer_details.clear
  puts "Customer First Name:"
  $input = gets.chomp.to_s
  $input = $input.delete(' ')
```

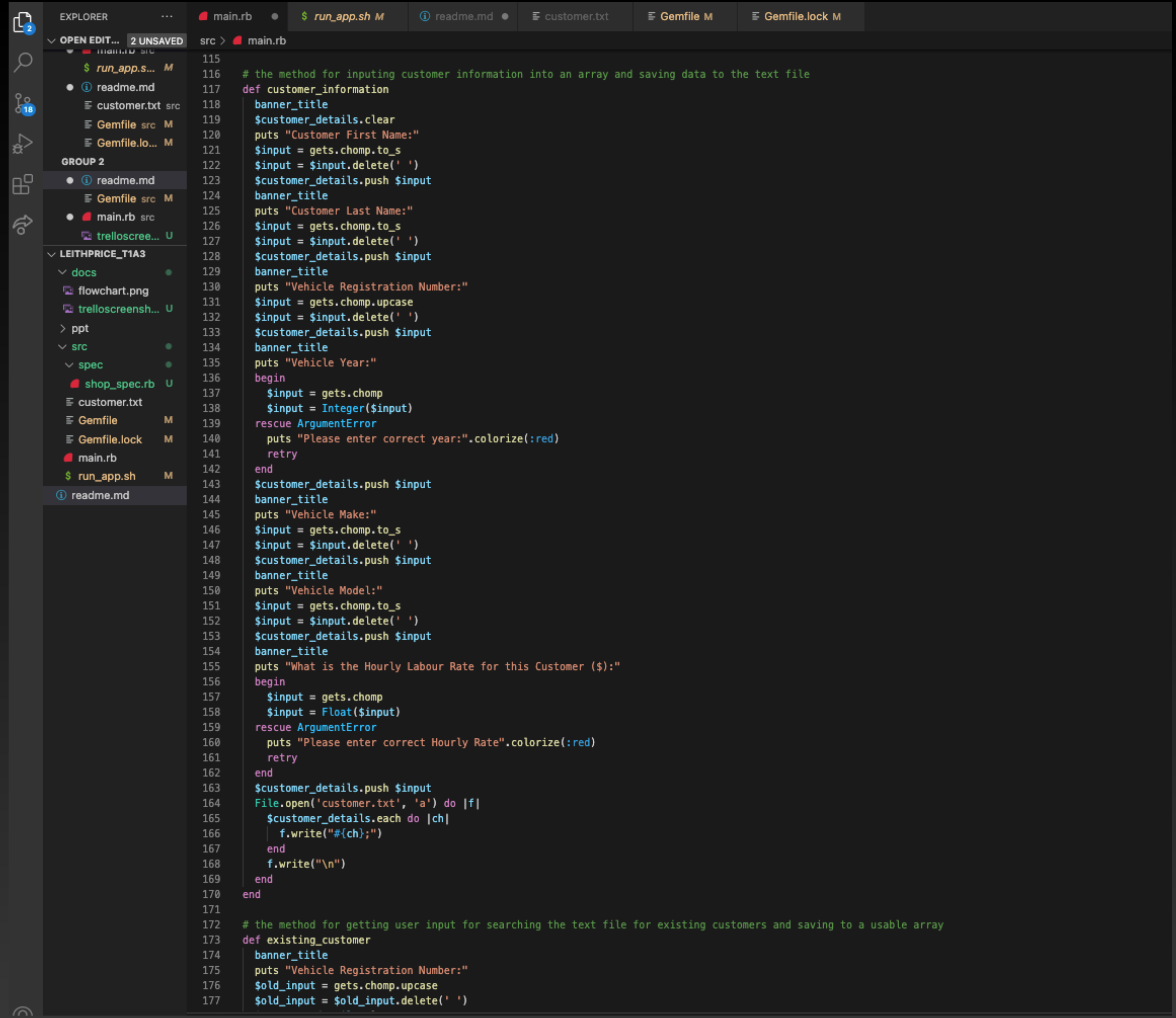
Program was made up of multiple methods.

Methods for

Displaying the invoice

Accepting labour time

Accepting odometer

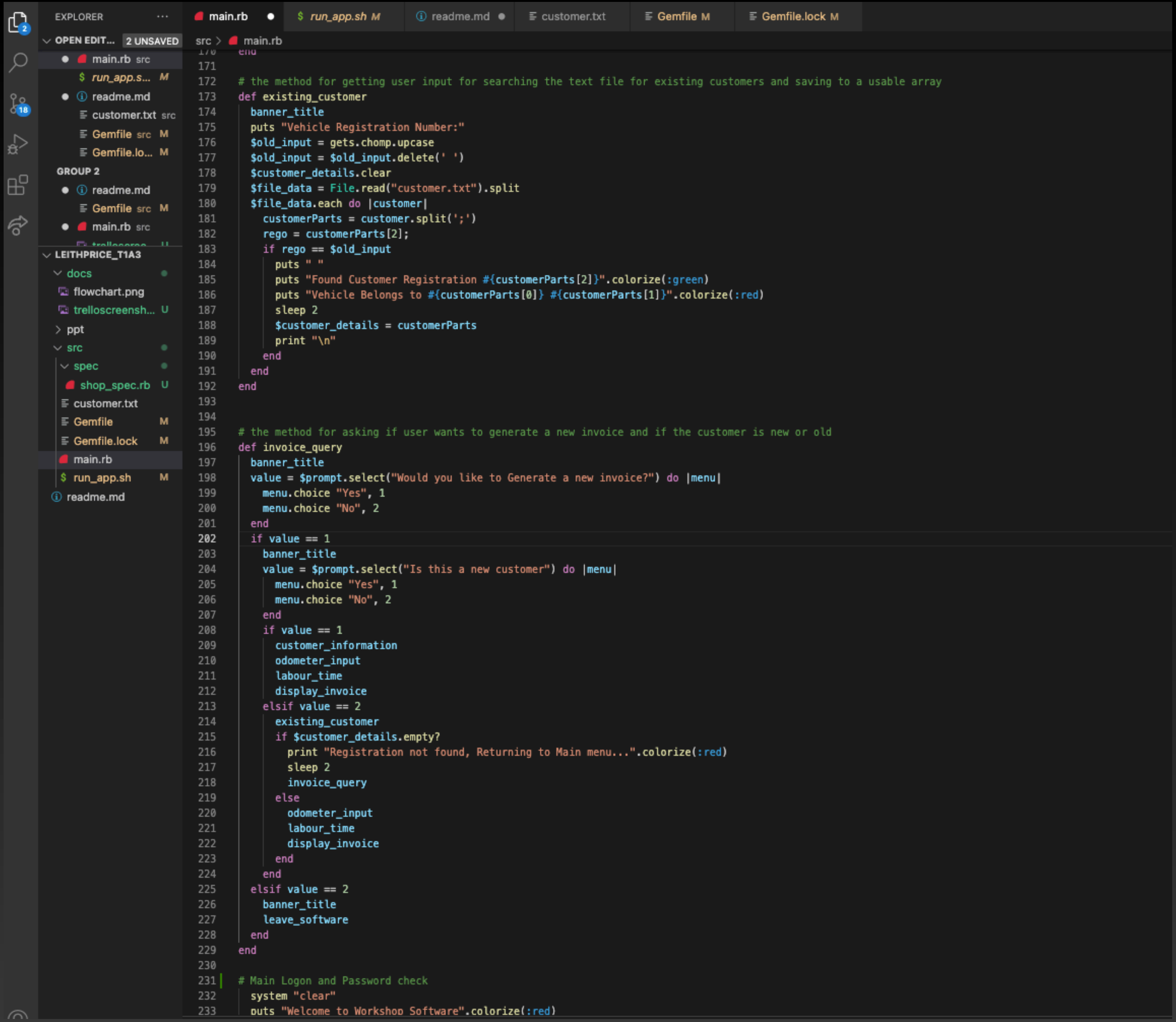


```ruby

# the method for inputing customer information into an array and saving data to the text file
def customer_information
  banner_title
  $customer_details.clear
  puts "Customer First Name:"
  $input = gets.chomp.to_s
  $input = $input.delete(' ')
  $customer_details.push $input
  banner_title
  puts "Customer Last Name:"
  $input = gets.chomp.to_s
  $input = $input.delete(' ')
  $customer_details.push $input
  banner_title
  puts "Vehicle Registration Number:"
  $input = gets.chomp.upcase
  $input = $input.delete(' ')
  $customer_details.push $input
  banner_title
  puts "Vehicle Year:"
  begin
    $input = gets.chomp
    $input = Integer($input)
  rescue ArgumentError
    puts "Please enter correct year:".colorize(:red)
    retry
  end
  $customer_details.push $input
  banner_title
  puts "Vehicle Make:"
  $input = gets.chomp.to_s
  $input = $input.delete(' ')
  $customer_details.push $input
  banner_title
  puts "Vehicle Model:"
  $input = gets.chomp.to_s
  $input = $input.delete(' ')
  $customer_details.push $input
  banner_title
  puts "What is the Hourly Labour Rate for this Customer ($):"
  begin
    $input = gets.chomp
    $input = Float($input)
  rescue ArgumentError
    puts "Please enter correct Hourly Rate".colorize(:red)
    retry
  end
  $customer_details.push $input
  File.open('customer.txt', 'a') do |f|
    $customer_details.each do |ch|
      f.write("#{ch};")
    end
    f.write("\n")
  end
end

# the method for getting user input for searching the text file for existing customers and saving to a usable array
def existing_customer
  banner_title
  puts "Vehicle Registration Number:"
  $old_input = gets.chomp.upcase
  $old_input = $old_input.delete(' ')
```

Program was made up of multiple methods.

Methods for

Collecting and saving the customer info



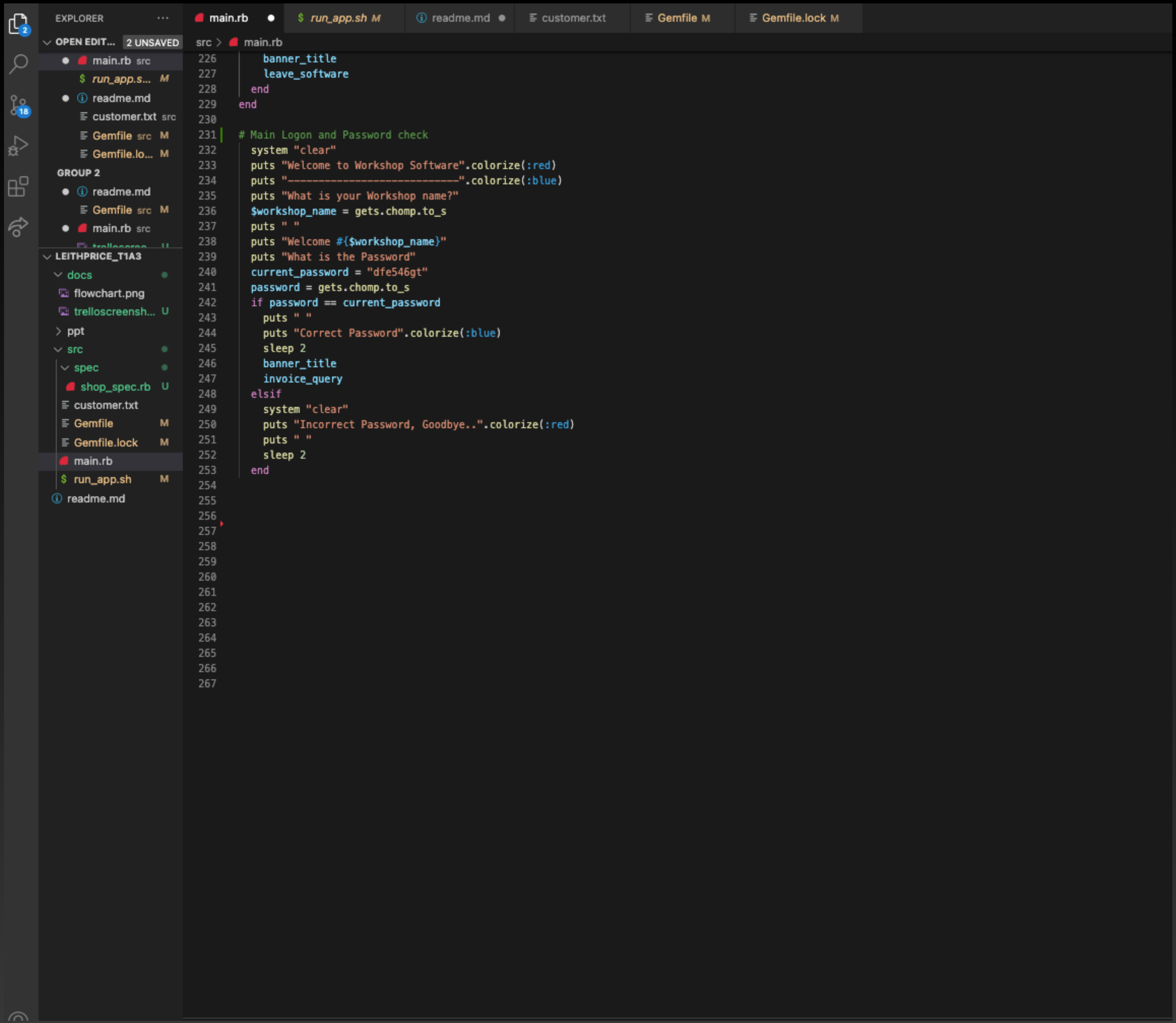
```ruby
end

# the method for getting user input for searching the text file for existing customers and saving to a usable array
def existing_customer
  banner_title
  puts "Vehicle Registration Number:"
  $old_input = gets.chomp.upcase
  $old_input = $old_input.delete(' ')
  $customer_details.clear
  $file_data = File.read("customer.txt").split
  $file_data.each do |customer|
    customerParts = customer.split(';')
    rego = customerParts[2];
    if rego == $old_input
      puts " "
      puts "Found Customer Registration #{customerParts[2]}".colorize(:green)
      puts "Vehicle Belongs to #{customerParts[0]} #{customerParts[1]}".colorize(:red)
      sleep 2
      $customer_details = customerParts
      print "\n"
    end
  end
end


# the method for asking if user wants to generate a new invoice and if the customer is new or old
def invoice_query
  banner_title
  value = $prompt.select("Would you like to Generate a new invoice?") do |menu|
    menu.choice "Yes", 1
    menu.choice "No", 2
  end
  if value == 1
    banner_title
    value = $prompt.select("Is this a new customer") do |menu|
      menu.choice "Yes", 1
      menu.choice "No", 2
    end
    if value == 1
      customer_information
      odometer_input
      labour_time
      display_invoice
    elsif value == 2
      existing_customer
      if $customer_details.empty?
        print "Registration not found, Returning to Main menu...".colorize(:red)
        sleep 2
        invoice_query
      else
        odometer_input
        labour_time
        display_invoice
      end
    end
  elsif value == 2
    banner_title
    leave_software
  end
end

# Main Logon and Password check
  system "clear"
  puts "Welcome to Workshop Software".colorize(:red)
```

Program was made up of multiple methods.

Methods for

Searching for existing customer input

Main menu



```ruby
      banner_title
      leave_software
    end
  end

  # Main Logon and Password check
    system "clear"
    puts "Welcome to Workshop Software".colorize(:red)
    puts "----------------------------".colorize(:blue)
    puts "What is your Workshop name?"
    $workshop_name = gets.chomp.to_s
    puts " "
    puts "Welcome #{$workshop_name}"
    puts "What is the Password"
    current_password = "dfe546gt"
    password = gets.chomp.to_s
    if password == current_password
      puts " "
      puts "Correct Password".colorize(:blue)
      sleep 2
      banner_title
      invoice_query
    elsif
      system "clear"
      puts "Incorrect Password, Goodbye..".colorize(:red)
      puts " "
      sleep 2
    end
```

This was the main code and not a method

This was the main logon and password check

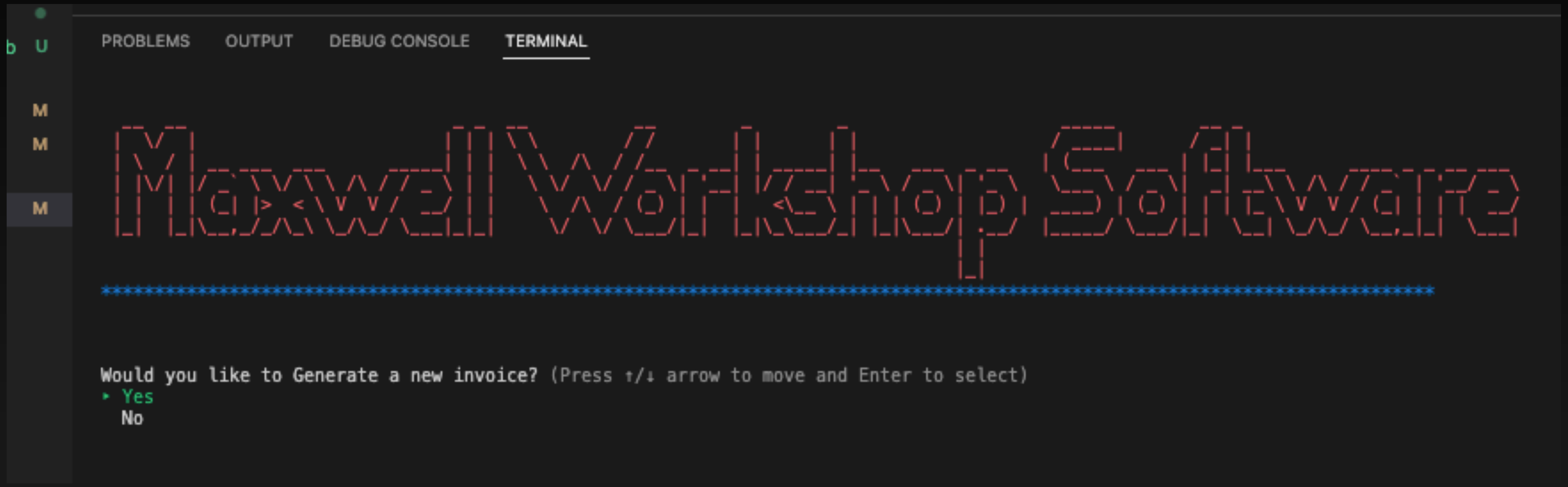
Main Features

- Password access limitations
- Invoice Display (print on screen)
- Customer details are saved in a .txt file
- Customer details can be searched for using the Registration no.

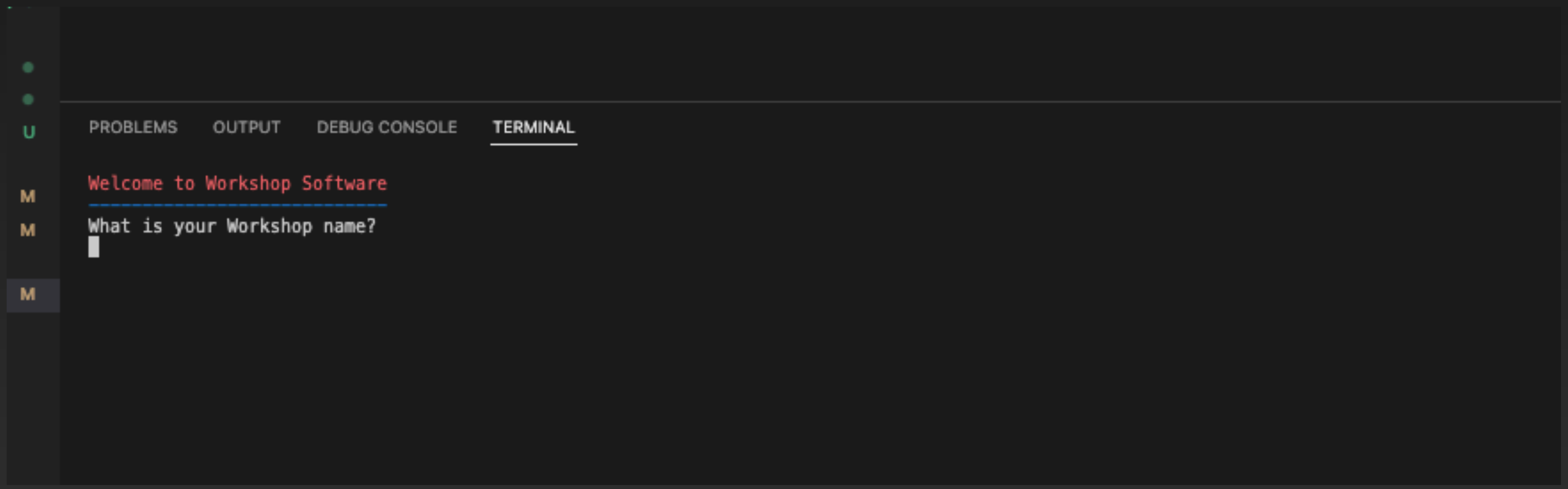
Gems used

- TTY - Prompt
- Colorize
- Artii
- TTY - Markdown
- RSpec

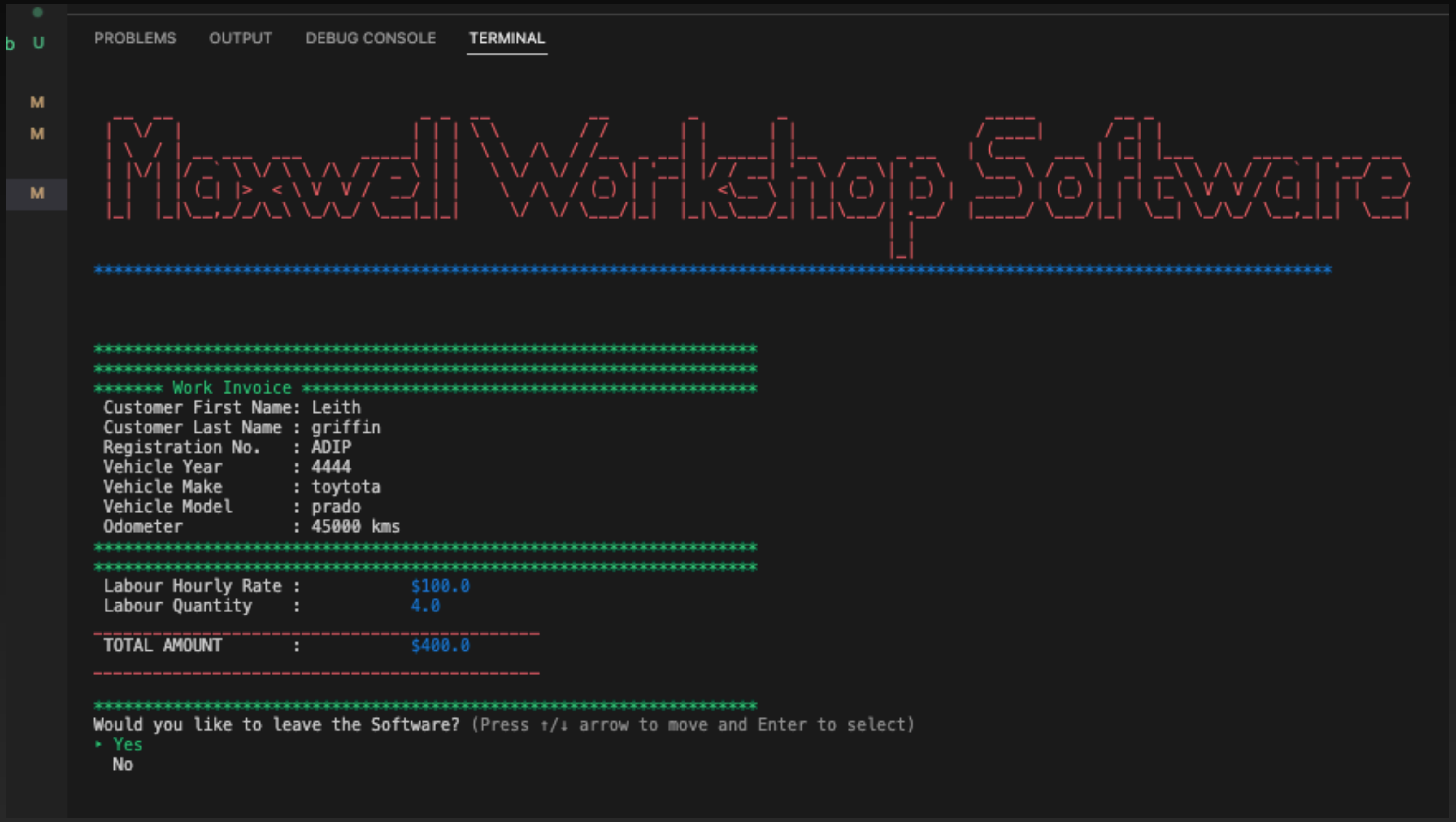
Example screenshots



```
******************************************************************************************************************************

Would you like to Generate a new invoice? (Press ↑/↓ arrow to move and Enter to select)
‣ Yes
  No
```



```
Welcome to Workshop Software
----------------------------
What is your Workshop name?
█
```



```
******************************************************************************************************************************

*******************************************************************
*******************************************************************
******* Work Invoice **********************************************
 Customer First Name: Leith
 Customer Last Name : griffin
 Registration No.   : ADIP
 Vehicle Year       : 4444
 Vehicle Make       : toytota
 Vehicle Model      : prado
 Odometer           : 45000 kms
*******************************************************************
*******************************************************************
 Labour Hourly Rate :          $100.0
 Labour Quantity    :          4.0
---------------------------------------------
 TOTAL AMOUNT       :          $400.0
---------------------------------------------

*******************************************************************
Would you like to leave the Software? (Press ↑/↓ arrow to move and Enter to select)
‣ Yes
  No
```